

A note on how the problem of Partion of Integers show in Caching

Mohit Thakur

Institute of Communications and Navigation
DLR (German Aerospace Center)
Oberpfaffenhofen-Wessling, Germany
Email: mohit.thakur@dlr.de

Abstract—In this article, we show how the finding the number of partitions of same size of a positive integer show up in caching networks. We present a stochastic model for caching where user requests (represented with positive integers) are a random process with uniform distribution and the sum of user requests plays an important role to tell us about the nature of the caching process. We discuss Euler's generating function to compute the number of partitions of a positive integer of same size. Also, we derive a simple approximation for guessing the guessing the number of partitions of same size and discuss some special sequences. Lastly, we present a simple algorithm to enumerate all the partitions of a positive integer of same size.

I. INTRODUCTION

Caching in networks is considered as an alternative to convert device memory into data rate to reduce delay and improve quality-of-service (QoS). The topic has been studied for a particular information theoretic setting in [1] and for decentralized setting in [2]. Since then, a lot of work has been done in coded-storage, device-to-device etc. with respect to caching. A good account of it can be found in [3] and [4].

In this article, we consider a user that uses a device to access content from the internet. The device is also referred to as a user equipment (UE). Suppose the device is connected to a server that can store (or cache) a limited amount of content in its memory. Such a server can be located within the UE or at a distant location as a separate entity. Figure II shows such a system with user requests arriving at UE that are partially or wholly served by the server S. Caching has many advantages such as improved quality of service (QoS). An important facet to study in this situation is how can we understand the nature of user requests for the content in relation to the cached content.

The problem can be modeled in a straightforward way using a stochastic setting. Let $F = \{F_1, F_2, \dots, F_N\}$ be a set of files with and $[K] = \{1, \dots, K\}$ be the set of first K positive integers. At a given instant the user requests a file from F at random. Suppose the stochastic process of user requests is given by $X = (X_1, X_2, X_3, \dots)$, where $X_k \in F$ and the probability that X_k is some file $F_j \in F$ is given by $P(X_k = F_j) = p_j$. An associated stochastic process $I = (5, 1, 7, 21, \dots)$ of file indices can be easily constructed from the user request process X , where the k^{th} element takes the value $j \in [N]$ if $X_k = F_j$. Now, if we ask the following questions of the following type



Fig. 1. Cache model.

- 1) what is the probability that a finite sub-sequence $\hat{I} \subset I$ sums to a positive integer M ?
- 2) does a finite sub-sequence exists such that the sum of it's elements equal's S ?

What the above questions have in common is that they require us to know the number of all possible partitions of integer $S = I_n + \dots + I_{n+k}$ into k parts such that each part belongs to $[N]$. In the next section we describe some classical results from the theory of partition of integers that gives a generating function for the problem.

II. GENERATING FUNCTION FOR INTEGER PARTITIONING

Let S be a positive integer that is a sum of k parts where each part belongs to $[N]$, i.e.,

$$S = r_1 + \dots + r_k,$$

where $r_j \in [N]$ for $1 \leq j \leq k$.

The coefficient of x^S in the Euler's generating function

$$\frac{1}{(1+x+x^2+\dots)(1+x^2+x^4+\dots)(1+x^3+x^6+\dots)\dots}$$

gives us the number of partition of S for all k . The general form of the Euler's generating function is given by

$$\mathcal{G}(x, y) = \frac{1}{(1-yx)(1-yx^2)(1-yx^3)\dots}.$$

Here, the coefficient of $x^S y^k$ is the total number of partitions of S in k parts. See sectio 3.16 in [5] for a detailed discussion.

Now we can play with some special cases that interests us. Consider two sets, U consisting of non-negative integers and V of positive integers. For instance, let $U = \{0, 1, 2, \dots, u\}$ and $V = [N]$. Then the generating function of the partitions of S in k parts belonging to the set V and allowed multiplicities

from U is given by

$$\mathcal{G}(x, y, U, V) = \prod_{v \in V} \left(\sum_{u \in U} y^u x^{uv} \right).$$

Even though $G(x, y, U, V)$ is a polynomial with finite terms, it is in general a NP-Complete problem to compute the coefficients of powers of xy . Next we present some special cases, when it is easier to approximate the coefficient of xy .

A. Sums with large k and large N

Consider a stochastic process I as described in section I and that each element is chosen uniformly from the set of positive integers \mathbf{Z}^+ . Let S_{n-m} denote the sum of elements

$$S_{n-m} = I_{m+1} + \dots + I_n.$$

The mean and variance of I_k are given by

$$\begin{aligned} \mu(I_k) &= \frac{1}{N} \sum_{i=1}^N i = \frac{N+1}{2} \\ \sigma^2(I_k) &= \frac{1}{N} \sum_{i=1}^N i^2 - \mu^2(I_k) = \frac{(8N+6)(N^2-1)}{24} \end{aligned}$$

Now, the central limit theorem tells that for any real $\alpha < \beta$

$$\begin{aligned} P \left(\alpha < \frac{S_{n-m} - (n-m)\mu(I_k)}{\sigma(I_k)\sqrt{n-m}} < \beta \right) \\ \rightarrow \mathfrak{N}(\beta) - \mathfrak{N}(\alpha), \end{aligned}$$

as $n-m$ goes to infinity. Therefore, we have

$$\begin{aligned} P \left((\alpha\sigma(I_k)\sqrt{n-m} + (n-m)\mu(I_k)) \right. \\ \left. < S_{n-m} < (\beta\sigma(I_k)\sqrt{n-m} + (n-m)\mu(I_k)) \right) \\ \rightarrow \mathfrak{N}(\beta) - \mathfrak{N}(\alpha) \end{aligned} \quad (1)$$

as $n-m$ goes to infinity. Let $\mathcal{N}(S, k, U, [N])$ denote the number of partitions of S in k parts, where each part belongs to $[N]$ with multiplicities from U . Then from inequality 1 we get

$$\mathcal{N}(S, k, U, [N]) \approx \frac{(n-m)!}{\left\lfloor \frac{n-m}{2} \right\rfloor!} \frac{1}{P(\alpha, \beta, S_{n-m})}, \quad (2)$$

where $P(\alpha, \beta, S_{n-m})$ denotes the probability

$$\begin{aligned} P \left((\alpha\sigma(I_k)\sqrt{n-m} + (n-m)\mu(I_k)) \right. \\ \left. < S_{n-m} < (\beta\sigma(I_k)\sqrt{n-m} + (n-m)\mu(I_k)) \right). \end{aligned}$$

Remark: The approximation 2 calls for the following remarks.

- 1) Approximation 2 is bounded by the error term that only vanishes as $(n-m)$ tends to infinity.
- 2) The first term of right hand side of approximation 2 accounts for that for large sums with large $(n-m)$, on the average $(n-m)/2$ elements tends to be the same. Hence, the term accounts for the total number of permutations for a partition.

B. Special Sums

Let the sub-sequence of elements be of length m with k elements of value $N-1$ and $m-k$ elements with value N , where $0 \leq k \leq m$. Then the sum

$$S = k(N-1) + (m-k)N = mN - k$$

cannot be partitioned in m parts using any other element from $[N]$, other than $N-1$ and N . Therefore, there exist only one partition S with m parts.

Furthermore, as elements of the sub-sequence take the values uniformly and randomly from $[N]$, the sub-sequence can occur in

$$\frac{m!}{k!}$$

ways. Therefore, the probability of a sub-sequence that is m elements long and sums to S is

$$\frac{k!}{m!}.$$

III. ENUMERATION OF PARTITIONS OF SAME SIZE

In this section we present a simple algorithm to enumerate the partitions a positive integer into m parts.

Consider again a positive integer S that is to be divided in m parts, where each part belongs to the set $[N]$. And let $\mathcal{N}(S, k, [N])$ denote the total number of such partitions. Then we have

$$S = k \cdot d + r.$$

Here, $0 \leq r < d$ and $d \in [N]$. This means the S can be written as a sub-sequence

$$\underbrace{(d, \dots, d)}_{\hat{k}}, \underbrace{(d+1, \dots, d+1)}_{\hat{r} = m - \hat{k}}, \quad (3)$$

where all the elements sum up to S . We can rewrite the sub-sequence as follows

$$\underbrace{(d_1, \dots, d_1)}_k, \underbrace{(d_2, d_3, \dots, d_3)}_r, \quad (4)$$

where d_1, d_2 and d_3 are chosen according to

- 1) Decrease the left most element that is greater than 1 and increase the right-most element that is less than N , until the sequence becomes either $(1, \dots, d)$, $(1, \dots, 1, d, N, \dots, N)$ or $(N-1, \dots, N-1, N, \dots, N)$, where $1 \leq d \leq N$. Of course, for the third case there is only sequence that partitions S .

From the sequence 4, another sequence of the same sum can be constructed by simply by reducing right-most r elements by 1 and distributing the value r to the rest of the elements by increasing the elements up to $d_3 - 1$ starting from the k^{th} , then $k-1^{th}$ element and so on. For instance, the sequence $(1, 1, 1, 5, 8, 8)$ has $k = 3$ and $r = 2$, and after the process it becomes $(1, 1, 1, 7, 7, 7)$ while the sum of two

sequences remain the same. Denote the values of the i^{th} element sequence by e_i and let

$$\hat{r} = \sum_{j=1}^{m-r} (e_m - 1 - e_j)$$

and $\hat{m} = \begin{cases} m & \text{if } \hat{r} - r \geq 0, \\ m - (\hat{r} - r) & \text{otherwise.} \end{cases}$

If $\hat{r} - r = -r' < 0$, then the process can be applied to the first $\hat{m} = m - r'$ elements of the sequence while keeping the last r' elements the same. Otherwise, \hat{m} equals m . Let us call this process as the *enumeration process*. The following algorithm describes the enumeration process for a given \hat{m} .

Algorithm 1 Enumeration Process

Input: $S = \{(d_1, \dots, d_1, d_2, d_3, \dots, d_3)\}$, r and \hat{m}

Output: $\mathcal{N}(S, m, [N])$

```

1: Evaluate  $\hat{r} = \sum_{j=1}^{m-r} (e_m - 1 - e_j)$ 
2: repeat
3:   if  $\hat{r} - r \geq 0$  then
4:     for  $l = 1$  to  $\hat{m} - r - 1$  do
5:        $e_{m-r-1} = \min(e_{m-r-1} + r, e_m - 1)$ 
6:        $e_{m-r-l} = \min(1 + [r-l(e_m-1) - e_{m-r-1}], e_m - 1)$ 
7:        $e_{m-r+1} = \dots = e_{m-r+\hat{m}} = e_m - 1$ 
8:        $\mathcal{N}(S, m, [N]) = \mathcal{N}(S, m, [N]) + 1$ 
9:        $r \leftarrow \min\left(\hat{m}, r + \left\lfloor \frac{r}{d_3-1-d_2} \right\rfloor + \left\lfloor \frac{r-(d_3-1-d_2)}{d_3-1} \right\rfloor\right)$ 
10:       $\hat{m} \leftarrow \sum_{i=1}^m (e_m - e_i)$ 
11:      return  $s_i = (e_1, \dots, e_m)$ 
12:       $\mathcal{S} \leftarrow \mathcal{S} \cup s_i$ 
13:    end for
14:  else
15:     $m \leftarrow m - (\hat{r} - r - 1)$ 
16:    go to 1
17:  end if
18: until  $e_r - e_{r-1} = 1$ 
19: return  $\mathcal{S}$ 

```

If we repeat this process for $\hat{m} \in [m]$, we get the algorithm that enumerates all partitions of S that have m parts each.

Algorithm 2 Enumeration algorithm for integer partitions of same size

Input: \mathcal{S}^m and m

Output: $\mathcal{N}(S, m, [N])$

```

1: for  $\hat{m} = m$  to  $\hat{m} = 2$  do
2:   Enumeration Process( $S$ )
3: end for
4: return  $\mathcal{N}(S, m, [N]) = |\mathcal{S}^m|$ 

```

It is straightforward to see that the algorithm 2 is exhaustive and enumerates all the partitions of S of size m while counting

the number of partitions. We leave out the proof out for brevity. In general, the enumeration algorithms do not serve much purpose in determining a useful formula for computing the number of partitions except for the special case when the problem instances are really small, but they can be a source of insight to the nature of the problem.

REFERENCES

- [1] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, pp. 2856–2867, 2014.
- [2] —, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw.*, vol. 23, pp. 1029–1040, 2015.
- [3] G. C. Mingyue Ji and A. F. Molisch, "Fundamental limits of caching in wireless d2d networks," *IEEE Trans. Inf. Theory*, vol. 62, pp. 849–869, 2016.
- [4] E. B. Georgios Paschos, I. Land, G. Caire, and M. Debbah, "Wireless caching: Technical misconceptions and business barriers," *arXiv*, 2016.
- [5] H. S. Wilf, *Generatingfunctionology*. San Diego: Academic Press, Inc., 1994.